DOCUMENT RESUME

ED 392 458 IR 055 843

AUTHOR Ghanti, Srinivas; Evans, John E.
TITLE Method of Coding Search Strings as Markov Processes Using a Higher Level Language.
PUB DATE 95
NOTE 26p.
PUB TYPE Reports - Descriptive (141)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS Coding; *Markov Processes; Mathematical Models; *Online Searching; *Predictive Measurement; Probability; Programming; *Search Strategies; *Statistical Analysis
IDENTIFIERS *Query Processing; Variables

ABSTRACT

For much of the twentieth century, Markov theory and Markov processes have been widely accepted as valid ways to view statistical variables and parameters. In the complex realm of online searching, where researchers are always seeking the route to the best search strategies and the most powerful query terms and sequences, Markov process analysis can investigate interactive search input variables in order to identify preferred search algorithms. This report uses the Statistical Analysis System (SAS) to demonstrate the development of a higher-level language coding for these processes. Utilizing the power of this widely available statistical package should facilitate the application of this technique for similar analyses of sequence-based activities. The discussion includes identification of applications in information retrieval analysis and common problems in statistical and analytical technique. Includes relevant output in the form of three figures and four tables. An appendix gives instructions on how to program workable code for Markov string analysis using the SAS. (Contains 36 references.) (BEW)

# Method of Coding Search Strings as Markov Processes

# Using a Higher Level Language

Srinivas Ghanti and John E. Evans*

University of Memphis Libraries

Memphis, Tennessee U. S. A. 38152-1000

## Abstract:

The development of a higher-level language coding for Markov process analysis of interactive search input variables is demonstrated using the Statistical Analysis System (SAS). Utilizing the power of this widely-available, higher-level statistical package should facilitate the application of this technique for similar and related analyses of sequence-based activities. The discussion includes identification of applications in information retrieval analysis, common problems with statistical and analytical technique, samples of workable code, and examples of relevant output.

# Method of Coding Search Strings as Markov Processes

# Using a Higher Level Language

Srinivas Ghanti and John E. Evans*

University of Memphis Libraries

Memphis, Tennessee U. S. A. 38152-1000

*Address all correspondence and inquiries to:

| | |
|---|---|
| Dr. John E. Evans | Voice: 901-678-4485 |
| University of Memphis Libraries | FAX: 901-678-8218 |
| Memphis, TN 38152 U. S. A. | E-Mail: evansje@cc.memphis.edu |

Acknowledgment: This work was funded, in part, by a Faculty Research Grant from the University of Memphis. Sincere appreciation is expressed to Dr. Donald R. Franceschetti, Vice Provost for Research, for his interest and support.

# Method of Coding Search Strings as Markov Processes

# Using a Higher Level Language

Srinivas Ghanti and John E. Evans*

University of Memphis Libraries

Memphis, Tennessee U. S. A. 38152-1000

## Introduction, Background, and Purpose:

In the past, applications of *Markov processes* have been conducted involving the statistical analysis of patterns of events in a variety of disciplines. Feller (1950) provides a classic introduction to the area of applied probability. Bharucha-Reid (1960) and Bhat (1984) also illustrate applications especially useful and effective in biology, while Chang (1968) focuses almost exclusively on biostatistics. Early applications to computer science were provided by Kleinrock (1976) and Allen (1978) followed by Trivedi (1982). Unique studies were conducted in geology by Krumbein & Dacey (1969) and Schwarzacher (1969), in part, using reverse analyses with special utility in that discipline. Excellent, highly-readable introductions to this area of probability analysis are to be found in the works of Takacs (1960 & 1967) and Kemeny & Snell (1960).

Historically, Markov theory can be traced to the early works of Markov (1907). Its popularity and application, supplemented by notable theoretical advances may be attributed to Kolmogorov (1931 & 1936). In each of these applications, a group of variables, associated by a known parameter, such as time or space, is considered a stochastic process and may describe random phenomena. Such sequences are known as state strings and are considered *Markov* *processes* if the conditional distribution is dependent only on the most recent known value of the process. Further, when these processes are characterized by a discrete parameter space, *I. e.*, a finite, determinable number of variables, the term *Markov chain* is used.

A Markov state transition of order 1 is the probability of change between two states. A state transition of is *k*th order where $k$ => $^{1}$ and $k = n - 1$ where $n$ is the sum of discrete states under investigation.. For each model, traditionally, *transition probability matrices* (TPM) are calculated and tested, using the *Kolmogorov-Smirnov (K-S)* test of hypothesis, against the probabilities generated using the *Markov chain analysis*. The results of these analyses provide measurable probabilities of state transition strings, or *Markov chains*, of varying lengths $n$ or order $k$ where $k=n-1$ as noted above. That is, what is the probability that state $b$ will precede or follow state $a$. A *Markov analysis* predicts the probability of these changes from $a$ to $b$ or conversely.

The method described in this contribution is a simplified process, the power of the method reported here is that it does not require the theoretical analysis as found with *K-S* or other mathematical techniques. This process directly generates a statement of state transition probabilities. Application of this routine should greatly increase the use of *Markov analyses* by overcoming the mathematical and statistical impediments to research. This program, and its specific adaptations, will serve as the foundation for countless applications involving the analysis of sequential activities, greatly facilitated by this powerful routine.

In information science, one of the earliest references to *Markov processes* is found in Weaver (1949) wherein he makes the theoretical and, ultimately, practical transition from a symbol sequence to a stochastic process, to a Markov process or chain, thence to ergodic processes which "tend to be representative of the sequence as a whole . . . (and) exhibit a particularly safe and comforting sort of statistical regularity" (p. 102). In the field of information science, these techniques are particularly useful in analyzing how individuals, regardless of skill level or experience, conduct interactive database searches and what procedures are used in these activities; such inquiries should lead to a better understanding of how users interact with the system. This analytical process gives valuable information on system and user performance by describing and analyzing the sequence of events used in actual activities not as they might be theoretically predicted. These analyses can be used to improve the system design and the user interfaces. For such an analysis, the use of a stochastic process analysis model seems to be the most effective technique. In this methodology, the user commands are mapped onto a set of numerical codes to generate what are called the *state strings* or, in our case, *input variable strings (IVS)*, noting the specific applications to database search input.

*Markov processes* are described by several characteristics; of particular interest to information retrieval research is the accessibility of one state from another, the transient nature of the states, and their non-recurrent state which means that the terminus is not identical to the starting point. To say that the variable states are accessible to one another is to say that they *communicate*, meaning that from any state, all other states are equally and directly *accessible*. From Markov theory and general mathematics we understand that this condition entails characteristics of reflexivity, symmetry, and transitivity; thus, the sets of states form an *equivalence* class. If all states within the set belong only to that set, the set is *irreducible*. More generally, a characteristic of a Markov state is also a characteristic of the class; these attributes cannot be otherwise. A final, relevant property is that of *transience*; a *Markov chain* will either return to its original state or will progress to infinity (theoretical) or to some other non-initial state. In this application to information science, namely, interactive database searching, the clear intent, and overwhelmingly common practice is to end somewhere other than the point of origin.

Findings to date, as reported in the various works by Fenichel (1980), Penniman (1975 & 1982), Penniman and Dominick (1980), Tolle & Hah (1985), Cooper (1983), Chapman (1981), and Zink (1991) furtner suggest that applicable string sequences in information retrieval are *aperiodic*, there being no discernible (or reported) repetition of sequences. It is more than a suggestion that there appears to be a wide variety of techniques, internal stopping rules, and cognitive representations of the practice of interactive searching (Standera, 1975; Penniman, 1975; Wildemuth, Jacob & Fullington, 1991). An early, but now dated review of the relevant research was provided by Fenichel (1981) setting the stage for the subsequent decade of continued research. Though much research in this area has focused on the circumstances surrounding bibliographic databases, concomitant difficulties are found in full-text databases as ably described by Tenopir (1985). Lest some conclude that the emerging cyberspace of hypertext leaps and bounds will eliminate the need for structural organization of search activity, the research of Pollard (1993) should serve to dissipate that misconception.

Therefore, *Markov chains*, as used in information retrieval environments, are *communicative, equivalent, irreducible, non-recurrent, aperiodic, and transient*. This we know to characterize the conditions of information retrieval. The remaining characteristic of the class and the states is whether search string variables are ergodic as Weaver (*supra*) would suggest. Therein lies the fundamental research question to be addressed elsewhere (Evans & Ghanti) and that has been addressed by others in the aforementioned works; what is the predictable (statistically drawn) nature of the search process? Previous Markov analyses of interactive search technique have been presented with useful results (see Penniman (1982), Chapman (1981), and Pao & McCreery (1986) for significant contributions), though the elaboration of the results has not benefitted from the fullness of exposition afforded by the current computational technique which we seek to make available. These reports also discuss observational technique and analysis as do Brown & Agrawala (1974), Dominick & Penniman (1979), Chapman (1981), Penniman (1975 & 1977), Penniman & Dominick (1979 & 1980) and Wanger, McDonald & Berger (1980).

**Problem:**

Online interactive searching of text or bibliographic databases involves a deceptively complex array of behaviors representing the logical, machine-focused representation of a researcher's intellectual problem represented in natural language, using a standard query language, supplemented by specialized, subject-specific (thesaurus-based) terminology. These representations are initially and commonly illustrated, simply enough, by Boolean logic and Venn diagrams or some variation thereof. While these methods may represent a convenient and simplified visual relationship of several key terms before the initiation of the searching process, such illustrations do not represent the complexity of the sequence of events that the searcher will use to satisfy the intellectual query as mediated by human understanding and electronic protocols in the interactive process. Indeed, any sophisticated information retrieval (IR) system will provide a wide variety of powerful logical tools as part of its search engine; these may be used in any sequence. The challenge for effective file query has always been to find the best search strategy, the most powerful query sequence, and the most efficient protocol to produce the desired results. Applications of *Markov process analysis* allow investigations of successful searches to identify preferred search algorithms which lead to successful outcomes of the search.

Markov process representations and analyses provide a useful alternative to visual images representing, as they do, the dynamic transitioning of the search protocol over the common index parameter, in this case, time. The dynamic environment has been the most illusive illustration of interactive domains. Boolean logic and Venn diagrams images are inherently *a priori*. Once the online search is underway, the illustration of the dynamic environment can approach chaos as terms and relationships are explored, searched, and connected.

When analyzing systems suspected of Markov-like stochastic processes, the first activity is the classification of states. The analysis proceeds to study different order, *k*, (string lengths minus one) models separately. Using a simple coding scheme to identify the range of actions chosen by a searcher, the search process may be represented easily by the coded string; subsequently, this coded data can then be subjected to *Markov analysis*. The result of such representations should be the identification and representation of patterns of interactive query behavior: these may be specific to subject disciplines; or they may be related to the individual searchers as extensions of their cognitive processes; or they may represent more generalized patterns of user-machine interaction; or we may find that revealed patterns are completely random (if anything is still random!).

Several useful coding schemes for online interactive search actions and behavior have been developed; particularly noteworthy are the contributions of Tolle & Hah (1985), Penniman (1982), Chapman (1981), Bates (1979), and Fidel (1985). Our project used a slight variation of these coding schemes. which was designed to be most applicable to our research and the best fit with our principal search system (BRS). The details of this application will be found elsewhere (Evans & Ghanti).

## Methods and Procedures

Having collected the data and developed a coding scheme, the next step is to assign a code to each step of the search session. Sample search statements were coded using a scheme which provided a numerical code to represent the interactive actions of the online searcher. This data coding was then entered into a database for subsequent analysis. Each search was identified by its session log number. This is illustrated with brief explanations of the search action taken, in the search session log example which appears as Figure 1. Having completed the coding of all searches the search sessions can be represented by the input variable string. Each search, for this portion of the analysis would be represented by a data record such as that shown in Figure 2. Once all the search session steps are coded, we can build a variable length array as shown for five sample searches represented by their *input variable strings* as in Figure 3. Each code number in the *input variable string* represents the *Markov state.* Of interest to this and related research is the probability of transitions from one state to another.

## Data Analysis for Markov Process

By subjecting this array to various analyses, we can get such information as a precise probability value that any specified string sequence of *n-order* will occur with probability *p*. The initial output of the program routine is a rank order frequency table indicating the sequence, its frequency of occurrence, and the cumulative frequency of all combinations to that point in ascending order. The frequency column itself is in descending order of frequency. In this limited, five-case example, we discover that in a first order state transition (*k=1, n=2*) with an initial *state* 7 (advanced term (thesaurus) search) will transit to *state* 8 (Boolean search) 13% (*p=0.13*) of the time. A second order (*k=2, n=3*) state transition combination of *state 6-7* transiting to *state 8* will occur 7.4% (*p=0.074*) of times. By direct observation, or unaided computational technique, these results are largely indeterminable. For human observation and calculation, even this simple set is daunting. Through the power of the SAS program described later (and provided in Appendix A), the user's effort is greatly reduced. Furthermore, the output file itself can then be subjected to additional statistical tests. Partial results for the first and second order *Markov chains* generated by this example are illustrated in Tables 1 and 2. From these Tables of results, we can build the *Transition Probability Matrices (TPM)* for the two state transition cases in this example as illustrated in Tables 3 and 4.

**SAS Programming**

The Statistical Analysis System (SAS) is used primarily for statistical analysis of numeric data and has many built-in functions to support this capability. SAS is very flexible and so it can be very effectively used, with smart programming, for the analysis of *Markov strings* and *processes* also. The powerful, functional capabilities and its widespread availability make SAS a preferred tool for this kind of analysis (as observed and predicted by Olsgaard and Evans as early as 1980 and mentioned by Williams (1982)). The SAS code used in our research is found in Appendix A.

**Common Problems**

The most common and troubling difficulty will be the extremely high demands on processor time and, especially, work space. As the character strings are parsed very large work space demands are placed on the processor. Our initial file space (a standard 10k blocks) was quadrupled before even fourth order processing could be completed. The files of coded data are rather small as files go;

however, the program demands enormous (by comparison) work space. Processor time can be significant as these things are measured. Each locality will have the solution to these problems.

Care must be taken in the *input variable string* to indicate the end of data for each search to avoid wrapping to the next line of the *IVS.* Not a matter of concern heretofore, we have not yet pursued the matter of generating the *Transition Probability Matrix (TPM)* directly from the output, relying on the comparatively simple transcription of the data as revealed in the resultant tables.

## Discussion

Despite the intricate demands of SAS coding and the need to overcome certain problems, the results provide a powerful demonstration of SAS, of *Markov analysis*, and, ultimately, the important and necessary analysis of search patterns. Direct observation, intuitive guesswork, or lengthy and tedious calculations can not equal the analytical power and vast data compilation needed. Our experience with 936 searches with search strings sometimes in excess of sixty search steps clearly demonstrates the power of this analytical engine. Details of this analysis will be provided separately.

## Acknowledgment:

Sincere appreciation is expressed to Computer Services at The University of Memphis for their kind cooperation and generous allowance of file space and processor times to complete these analyses. This work was funded in part, by a faculty research grant at The University of Memphis. Special appreciation is given to Dr. Donald R. Franceschetti, Vice Provost for Research for his interest and support.

## Figure 1: Search Example and Coding

Search ID: 5467 -- Unique search identifier.

| Search Statements | Code Assignments |
|---|---|
| 1_: (SERIALS or PERIODICALS).DE. | 8 -- Boolean search with field specification. |
| 2_: LIBRARY-AUTOMATION | 7 -- Advanced term (thesaurus) search. |
| 3_: 1 AND 2 | 8 -- Boolean combination. |
| 4_: CATALON | 6 -- Simple, single term search. |
| 5_: PURGE 4 | 3 -- Neutral, a command that does not directly affect structure. |
| 4_: CATALOG | 6 |
| 5_: ..P 4 TI/1-5 | 9 -- print (on screen). |
| 5_: ..PO 4 BIBL.AB.DOC=1-45/SORT=SO | 12 -- print (offline). |
| 5_: RETROCON | 6 |
| 6_: 5 NOT 4 | 8 |
| 7_: ..P 6 TI/I,DE/1 | 9 |
| 7_: RETROSPECTIVE ADJ CONVERSION | 7 |
| 8_: 1.MJ. AND 7 | 8 |
| 9_: 8 NOT 4 | 8 |
| 10_: ..O | 10 -- quit or logoff. |

## Figure 2 Search Coding Representation

| Search Session | Input Variable Coding |
|---|---|
| 5467 | 8 7 8 6 3 9 12 6 8 9 7 8 8 10 |

## Figure 3: Input Variable String Compilation

| **Session** | | **Search String Data** |
|---|---|---|
| | *cases=5* | |
| 5467 | | 8 7 8 6 3 9 12 6 8 9 7 8 8 10 |
| 5468 | | 6 7 8 7 6 8 9 12 10 4 7 8 9 12 10 |
| 5469 | | 7 7 6 7 8 9 10 |
| 5470 | | 6 7 8 3 6 6 7 8 12 10 |
| 5471 | | 7 7 10 |

## Table 1: First Order Markov State Transitions

| IVString | Frequency | Percent | Cumulative Frequency | Cumulative Percent |
|---|---|---|---|---|
| 7 8 | 7 | 13.0 | 7 | 13.0 |
| 10 . | 5 | 9.3 | 12 | 22.2 |
| 6 7 | 4 | 7.4 | 16 | 29.6 |
| 8 9 | 4 | 7.4 | 20 | 37.0 |
| 12 10 | 3 | 5.6 | 23 | 42.6 |
| 9 12 | 3 | 5.6 | 26 | 48.1 |
| . 6 | 2 | 3.7 | 28 | 51.9 |
| . 7 | 2 | 3.7 | 30 | 55.6 |
| 6 8 | 2 | 3.7 | 32 | 59.3 |
| 7 6 | 2 | 3.7 | 34 | 63.0 |
| 7 7 | 2 | 3.7 | 36 | 66.7 |
| 8 7 | 2 | 3.7 | 38 | 70.4 |

## Table 2: Second Order Markov States Transitions

| IVString | Frequency | Percent | Cumulative Frequency | Cumulative Percent |
|---|---|---|---|---|
| 6 7 8 | 4 | 7.4 | 4 | 7.4 |
| . 6 7 | 2 | 3.7 | 6 | 11.1 |
| . 7 7 | 2 | 3.7 | 8 | 14.8 |
| 10 . 6 | 2 | 3.7 | 10 | 18.5 |
| 10 . 7 | 2 | 3.7 | 12 | 22.2 |
| 12 10 . | 2 | 3.7 | 14 | 25.9 |
| 6 8 9 | 2 | 3.7 | 16 | 29.6 |
| 7 8 9 | 2 | 3.7 | 18 | 33.3 |
| 8 9 12 | 2 | 3.7 | 20 | 37.0 |
| 9 12 10 | 2 | 3.7 | 22 | 40.7 |
| 10 4 7 | 1 | 1.9 | 23 | 42.6 |
| 12 10 4 | 1 | 1.9 | 24 | 44.5 |

## Table 3: Transition Probability Matrix - First Order

*n=2, k=1*

| ↪ | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|
| **6** | 0.019 | 0.074 | 0.037 | | | | |
| **7** | 0.037 | | 0.13 | | | | |
| **8** | | 0.037 | | 0.074 | | | |
| **9** | | | | | | | 0.056 |
| **10** | | | | | | | |
| **11** | | | | | | | |
| **12** | 0.019 | | | | 0.056 | | |

## Table 4: Transition Proability Matrix - Second Order

*n=3, k=2*

| ↱ | 7 | 8 | 9 | 10 | 12 |
|---|---|---|---|---|---|
| 6-7 | | 0.074 | | | |
| 6-8 | | | 0.037 | | |
| 7-8 | 0.019 | 0.019 | 0.037 | | 0.019 |
| 8-9 | | | | | 0.037 |
| 9-12 | | | | 0.037 | |

**References:**

Allen, A. O. (1978) ***Probability, Statistics, and Queuing Theory***. New York: Academic Press.

Bharucha-Reid, A. T. (1960). ***Elements of the Theory of Markov Processes and Their Applications***. New York: McGraw-Hill.

Bhat, U. N. (1984). ***Elements of Applied Stochastic Processes*** 2nd ed. New York: Wiley.

Brown, R. N. & Agrawala, A. K. (1974) On the behavior of Users of the MEDLINE System, In Fenichel, C. H. (Ed.) ***Changing Patterns in Information Retrieval.*** Washington, D. C.: American Society for Information Science. 36-38.

Chapman, Janet L. (1981). A State transition Analysis of Online Information-Seeking Behavior. ***Journal of the American Society for Information Science. 32***, 325-333.

Chiang, C. L. (1968). ***Introduction to Stochastic Processes in Biostatistics***. New York: Wiley.

Cooper, Michael D. (1983) Usage Patterns of an Online Search System. ***Journal of the American Society for Information Science.*** 34, 343-349

Dominick, W. D. & Penniman, W. D. (1979) Interaction Monitoring Considerations Within Network-Based Information Systems. ***Collected papers of the 9th ASIS Mid-Year Meeting***, Pittsburgh, PA.

Evans and Ghanti *Analysis of Input-Output Variables in Online Bibliographic. . . .* (forthcoming)

Feller, W. (1950). ***An Introduction to Probability Theory and its Applications***. Vol. 1. (1st ed.); 3rd ed. 1968 New York: Wiley.

Fenichel, Carol Hansen. (1981). Online Searching: Measures that Discriminate among Users with

Different Types of Experiences. ***Journal of the American Society for Information Science***, 32, 23-32.

Kemeny. J. G. and Snell, J. N. (1960). ***Finite Markov Chains***. Princeton: Van Nostrand.

Kleinrock, L. (1976). ***Queuing systems (Vol. 2), Computer Applications***. New York: Wiley.

Kolmogorov, A. (1931) ***Mathematische Annalen***. 104, 415-458.

Kolmogorov, A. (1936) ***Matematicheskiui Sbornik***. N. S., 1, 607-610.

Krumbein, W. C. and Dacey, M. F. (1969) ***Journal of the International Association of Mathematics, Geology 1***, 1, 79-96.

Markov, A. A. (1907) ***Dynamic Prababilistic Systems.*** n.p. 552-576.

Mooers, Calvin N. (1960). The Next Twenty Years in Information Retrieval, some goals and predictions. ***American Documentation***, 11, 229-236.

Morehead, David R., and Rouse, William B. (1982). Models of Human Behavior in Information Seeking Tasks. ***Information Processing & Management***, 18(4), 193-205.

Pao, Miranda Lee, and McCreery, Laurie. (1986). Bibliometric Application of Markov Chains. ***Information Processing & Management***, 22(1), 7-17.

Penniman, W. D. (1975). A Stochastic Process Analysis of On-Line User Behavior. ***Proceedings of the 38th ASIS Annual Meeting,*** 12, 147-148.

Penniman, W. D. (1977). Suggestions for Systematic Evaluation of On-Line Monitoring Issues. ***Proceedings of the American Society for Information Science Annual Meeting***. 14, 241-249.

Penniman, David W. (1982). Modeling and Evaluation of On-Line User Behavior. ***Proceedings of the ASIS Annual Meeting,*** 19, 231-235.

Penniman, W. D. and Dominick, W. D. (1980) Monitoring and Evaluation of on-line information system usage. ***Information Processing and Management***. 16, 17-35.

Pollard, Richard (1993) A Hypertext-Based Thesaurus as a Subject Browsing Aid for Bibliographic Databases. ***Information Processing and Management***, 29, 345-357.

Schwarzacher, W. (1969). ***Journal of the International Association of Mathematics, Geology***, 1, 17-39.

Siegfried, Susan, Bates, Marcia J. & Wilde, Deborah N. (1993). A Profile of End-User Searching Behavior by Humanities Scholars: The Getty Online Searching Project Report No. 2. ***Journal of the American Society for Information Science***, 44 273-291.

Standera, Oldrich. (1975). On-Line Retrieval Systems: Some Observations on the User/System Interface. ***Proceedings of the 38th ASIS Annual Meeting***, 12, 38-40.

Takacs, L. (1960). ***Stochastic Processes***. New York: Wiley.

Takacs, L. (1967). ***Combinatorial Methods in the Theory of Stochastic Processes***. New York: Wiley.

Tolle, John E., and Hah, Sehchang. (1985). Online Search Patterns: NLM CATLINE Database. ***Journal of the American Society for Information Science***, 36, 82-93.

Trivedi, K. S. (1982). ***Probability and Statistics with Reliability, Queuing and Computer Science Applications.*** Englewood Cliffs: Prentice Hall.

Weaver, Warren (1949). ***The Mathematical Theory of Communication***. Urbana: The University of Illinois.

Wanger, J., McDonald, D. & Berger, M. (1980). ***Evaluation of the On-Line Search Process: A Final Report.*** Washington, D.C.: National Library of Medicine.

Wildemuth, Barbara M., Jacob, Elin K., and Fullington, Angela. (1991). A Detailed Analysis of End-User Search Behaviors [1]. ***Proceedings of the 54th ASIS Annual Meeting***, 28, 302-312.

Williams, Robert W. (1982). Non-Statistical Library Applications of the Statistical Analysis System (SAS). ***Proceedings of the American Society For Information Science Annual Meeting***, 339-341.

Zink, Stephen D. (1991, Spring) Monitoring User Search Success Through Transaction Log Analysis: The WolfPAC Example. ***Reference Services Review*** 19(1) 49-56.

# Appendix A

## Application Programming for Markov String Analysis

## Using The Statistical Analysis System (SAS)

```
/**************************************************************************

* Line numbers are used for illustrative reasons only. They should not be included in the program;

1    options ls=70;                    /* Specify the various options such as line size, page size,
                                       title, etc. at the start of the program.        */
2    data new;                         /* Create a new data set called "new" */
3    infile 'datafile.dat' missover;   /* Specify the data file name in single quotes. Thc missover
                                          option is used so that each line is treated as a separate
                                          record. */
4    input SesID +(-3) @;              /* Input the first four columns of each line as the unique
                                          Session ID. */
5    length IVCode 2;                  /* The length of the Input Variable Code is 2 in our case.
                                          This can be changed depending on the coding scheme
                                          being used. If using 3-digit codes, then this should be
                                          changed to 3. */
```

```
6       do until (IVCode= .);              /* Repeat until the end of the line. */
7               input IVCode @;            /* Read in the input variable codes into the successive
                                              variables IVCode */
8               output;                    /* Output it to the data set. */
9       end;
10      run;
11      data new;
12      length IVSTemp $24;    set new;   /* Select an appropriate size for the Variable string. */
/***    Lines 13-21: These lines generate the chains of the desired order. We are generating second
order chains in this example. For each Session ID, we generate as many chains as possible.   */
13      if SesID = lag(SesID) then do;
14              L2IVCode = lag2(IVCode);
15              L1IVCode = lag(IVCode);
16              IVSTemp  = L2IVCode  ||  L1IVCode  ||  IVCode;   /* Concatenates codes */
17              IVString  = compbl(IVSTemp);  /* Removes blank spaces between codes */
18      end;
19      else do;
20              IVSTemp = ' ';  IVString =  ' ';
21      end;
/***    In lines 14-17 numerals refer to the order of the Markov process.  For higher order analyses
these numerals must be sequentially incremented.  Additional lines may be necessary. */
22      proc freq
```

```
        order = freq;
        tables IVString;
    run;
    endsas;
/*****************************************************************/
```